Learning to extend legs of a transformable-wheel robot

James Clinton[1], Willy Li[1], Max Linden[2], Anthony J. Clark[2], and Elliot Hawkes[1]

[1]Department of Mechanical Engineering, University of California, Santa Barbara
[2]Computer Science Department, Pomona College

***Abstract*—Designing robots for dynamic, unstructured environments is a challenging task. Both the robot's mechanical design and control policy must be optimized for the environment. In this study, we explore the design of a transformable-wheel robot that can drive on smooth wheels and extend its legs to crawl over obstacles. We use reinforcement learning to train a policy for the robot that uses proprioception, namely motor velocity, body acceleration, and body orientation, to determine when to extend its legs. Our results indicate that this proprioception alone may be sufficient to control transformable wheels. In simulation, the robot was able to climb over a single step obstacle using only proprioception, though it was better able to do so with a collision sensor.**

I. Introduction

Robots are being designed for increasingly more complex environments. For example, uneven and unstructured terrain such as rocky hillsides, sand dunes, and forest floors. Much of the work in this area is focused on legged robots. In this study, however, we focus on transformable-wheel robots (see Fig. 1).

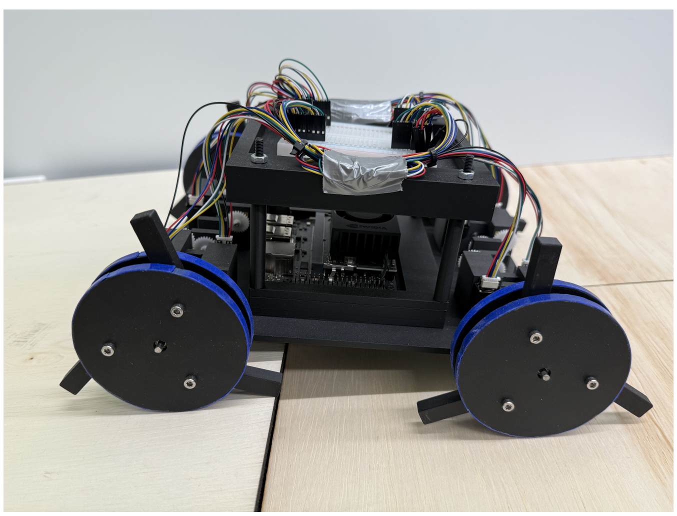


Fig. 1: Our transformable-wheel robot prototype, shown climbing up a step with its legs extended.

Transformable-wheel robots are able to both drive on regular, smooth wheels and also extend "legs" radially so that the robot can crawl over obstacles. Fig. 2 shows our robot in both configurations. These robots are inherently more stable than legged robots, as they do not require complex dynamically stable gaits for locomotion, and can shift to become even more stable in the wheeled state. They are also more capable than wheeled robots, with legs that enable them to traverse a more diverse set of terrain.

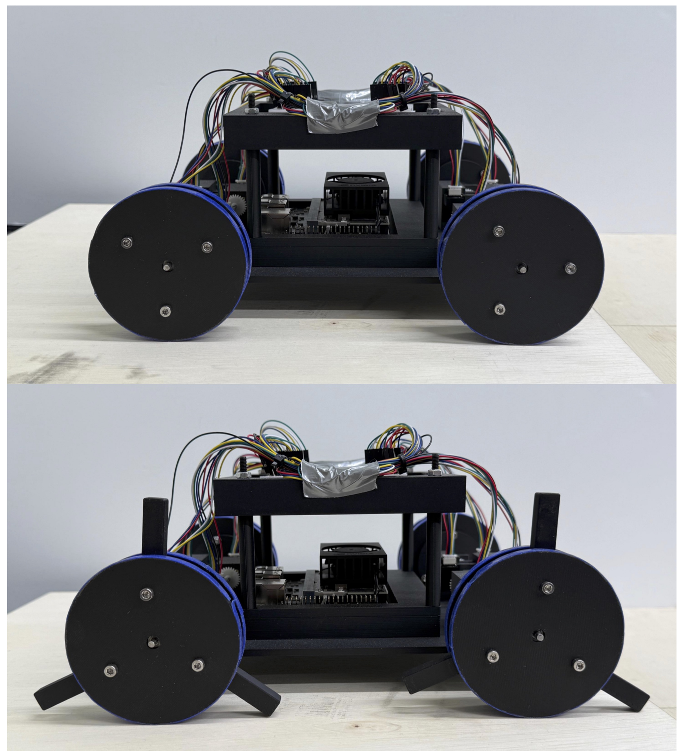


Fig. 2: A depiction of the transformable wheel with the leg extensions not extended (top) and extended (bottom).

The main challenge with transformable-wheel robots is in controlling wheel transformations. Specifically, deciding when and how far the robot should extend its legs. When the robot is on a relatively flat surface, it is more effective to drive on smooth wheels—there is no need to suffer the poorer sensor readings and increased wear and tear on the system in such cases. However, when the robot encounters an obstacle, it is important to extend its legs. The question is then: *how should the robot determine when to extend its legs, and what sensory input should it use?*

In this study, we propose a solution to this problem using reinforcement learning (RL). Our robot will use limited sensing—primarily proprioception—to determine when to extend its legs. Recent simulation packages (e.g., [12]) make it possible to train robots in a variety of configurations in a relatively short amount of time (i.e., hours or days instead of weeks), and recent advances in the area of sim2real make it more possible to transfer such policies to real-world robots [4].

We use PPO [10] to train a policy for the robot. The implementation is based on that provided by ManiSkill3, which is optimized to work with the accompanying simulation environment [12]. The policy controls both the speed of the wheel motors and the amount of leg extension.

In this study, we first use only proprioception. Transformable-wheel robots are more stable than legged robots, requiring less robustness from the controller. In addition, they provide good opportunities for inferring state information from proprioceptive data, as they are able to stably tilt, indicating the presence of obstacles or uneven terrain. This makes these robots a good fit for proprioception-based locomotion. To begin, our policy must learn to control the wheel motors and leg extensions using only encoder readings for each actuator and the robot's inertial measurement unit (IMU), which provides body acceleration and orientation data. Later, we include a simple collision sensor so that the robot knows when it is in contact with an obstacle.

The contributions of this work are a new transformable-wheel robot design, preliminary results of proprioceptive locomotion for that design, and an exploration of the sensor requirements for our transformable-wheel robot. Our results indicate that proprioception alone may be sufficient for controlling the transformable wheels.

II. Related Work

a) Transformable-Wheel Robots: Several studies have explored the design of transformable-wheel robots (a specific type of reconfigurable robots). Some have used unconventional methods of transforming between states. Chen et al. created a wheel that splits into two halves, aligning them to become a leg [2]. On the larger scale, Lee et al. developed a human-drivable vehicle with origami-based transformable wheels [6].

Many have used designs which incorporate leg extensions that fold out from a central wheel. Cao et al. developed a mechanism where each wheel "unfolds" into a legged-wheel [1]. Their design also includes rollers on each wheel similar to a Mecanum wheel. Similarly, Mertyüz et al. designed a wheel in which "fingers" fold in to form the wheel rim and out to act as a legged-wheel [14]. Zheng and Lee developed a similar wheel that passively actuates its legs in response to added load on the wheels [13].

Most studies in this area focus on the mechanical design of the robot, and less on decision making (i.e., when to transform the wheels). Our design is similar to past robots that use folding leg extensions, particularly [14], though our transformable wheel can rotate in both directions and is designed so that it can be quickly transformed to help the robot get unstuck. It also uses a differential coaxial shaft mechanism to drive its legs, differing from most past transformable wheels that directly drive their extensions, with a motor onboard the wheel.

b) Reinforcement Learning for Unconventional Mechanisms: Recently, some studies have explored the use of RL for controlling unconventional robots, including transformable-wheel robots. Park et al. developed an algorithm for transformable-wheel motion planning based on RL [8]. The algorithm is able to plan a transformation based on a model of the environment. Simohn et al. presented a energy-based reward function for learning how to control their RSTAR robot, which sprawls to control its center of mass and shape, so that it could climb over steps [11]. Chen et al. developed a wheel-legged robot that used RL to maneuver around obstacles using visual observations [3]. Unlike these studies, our focus is on simple proprioception.

c) Blind Locomotion: A recent body of work has explored the use of RL for controlling legged robots, particularly with proprioception. This blind locomotion relies on internal sensor measurements to the robot, rather than external observations of its surroundings, to navigate the world. For example, Lee et al. developed a blind locomotion policy for controlling a quadrupedal robot in diverse environments, highlighting the possibilities of using only proprioceptive sensor data [7]. Kumar et al. developed rapid motor adaptation algorithm on top of PPO to control a quadrupedal robot, robust to different environments as well as changing robot conditions, such as leg damage or added mass [5]. Their reward function is heavily based on the robot's cost of transport. Radosavovic et al. explored proprioceptive control for humanoid robots [9]. We aim to extend this work to an unexplored domain, transformable-wheel robots.

III. Methods

a) Physical Robot: The robot developed for this study is shown in Fig. 1. The robot chassis is $23\,\mathrm{cm}$ long and $16\,\mathrm{cm}$ wide. The robot has 4 wheels, each of which are $3.8\,\mathrm{cm}$ in radius $0.5\,\mathrm{cm}$ thick. Each wheel has three legs that can extend radially, with two wheel plates enclosing them. The wheels are lined with Dycem non-slip material to increase friction with the ground.

Each transformable wheel is driven by two motors, one that drives the wheel speed, and a second that controls the leg extension. Legs are "extended" (rotated) by differentially driving the two motors at different speeds. For example, if the leg motor is driven at a higher rate than the wheel motor, the three corresponding legs will gradually rotate outward. The motors are geared to coaxial shafts, enabling both motors to be placed on the body of the robot, rather than placing one on the wheel itself. Between the two plates of each wheel, the internal shaft of the leg extension motor is connected to a gearing system to drive the legs, shown in Fig. 3.

An encoder is attached to each motor (both the wheel and leg motors) for determining and controlling their speeds. Additionally, the robot is equipped with an inertial measurement unit (IMU) for measuring its orientation and acceleration.

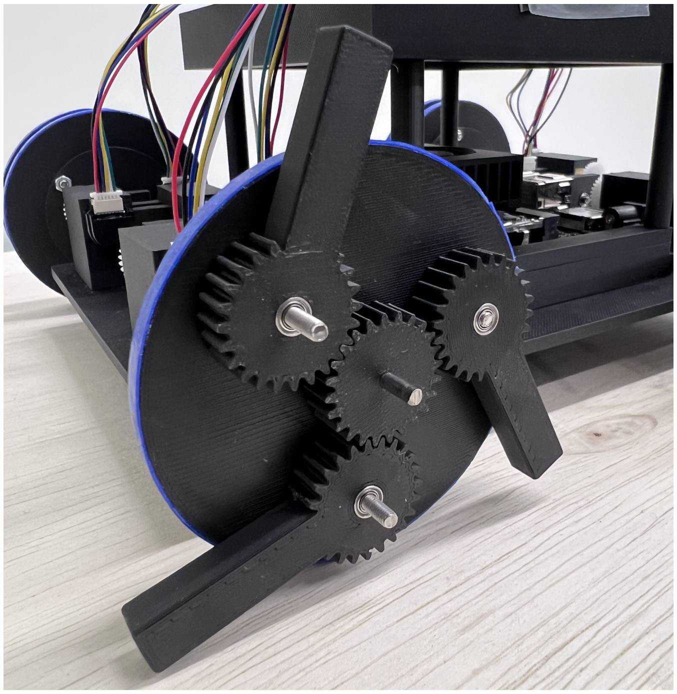


Fig. 3: The gearing mechanism of the transformable wheel. One motor is attached to the wheel, with the other attached to a coaxial shaft driving the legs.

In both simulation and on the real device, we measure the orientation, angular velocities, and linear accelerations, which are derived from IMU readings.

b) Simulation: Our simulation is based on the ManiSkill3 package [12]. The main advantage of this package is in its ability to simulate many different scenarios in parallel. Using an NVIDIA Tesla P100 (16GB), we are able to simulate close to 20 million steps per hour, where each step simulates $0.01\,\mathrm{s}$ for a single environment. We developed two custom environments (see Fig. 4) and modeled our robot using the package's facilities.

c) Reinforcement Learning: We are using a standard RL training loop for our experiments. The robot action space consists of the wheel and leg motor speeds (8 real-valued numbers). The observation spaces consists of the wheel encoder readings and IMU readings, i.e. orientation, angular velocities, and linear accelerations (18 real-valued numbers). The reward is a continuous function based on the robot's speed and the amount of leg extension.

$$r(t) = \frac{(1 - \tanh(5v_e(t))) + (1 - \tanh(e_t(t)))}{2}$$

where $r(t)$ is the reward at time t, $v_e(t)$ is the robot's velocity error (the different between the robot's current velocity and the desired velocity) and e_t is the amount of leg extension. The reward is normalized to be between 0 and 1, and the goal is to drive the robot at the desired speed while minimizing the amount of leg extension.

IV. Results

We trained policies for three different environments: **(1)** a flat plane with no obstacles, **(2)** a plane with a single step obstacle, and **(3)** a plane with a single step obstacle and a collision sensor.

The first policy (trained on a flat plane) will serve as a baseline for the other two. The policy was able to drive the robot at the desired speed without extending its legs and achieve a near perfect reward. The policy was then evaluated in the second environment (a plane with a single step obstacle), where it was unable to get past the obstacle.

The second policy (trained on a plane with a single step obstacle) was able to drive the robot at the desired speed and extend its legs when needed to climb over the obstacle. However, once the legs were extended, the robot did not learn to retract them. This is likely due to the noisy sensor readings caused by the robots extended legs, as well as the fact that the training episodes ended shortly after the robot overcame the step. We suspect that if the robot were trained over episodes longer than $4\,\mathrm{s}$, the policy would learn to retract the legs once the robot was over the obstacle.

The final policy (trained on a plane with a single step obstacle and a collision sensor) was able to drive the robot at the desired speed and extend its legs when needed. Moreover, it was able to partially retract its legs when they were not needed. This policy is shown controlling the robot in Fig. 4.

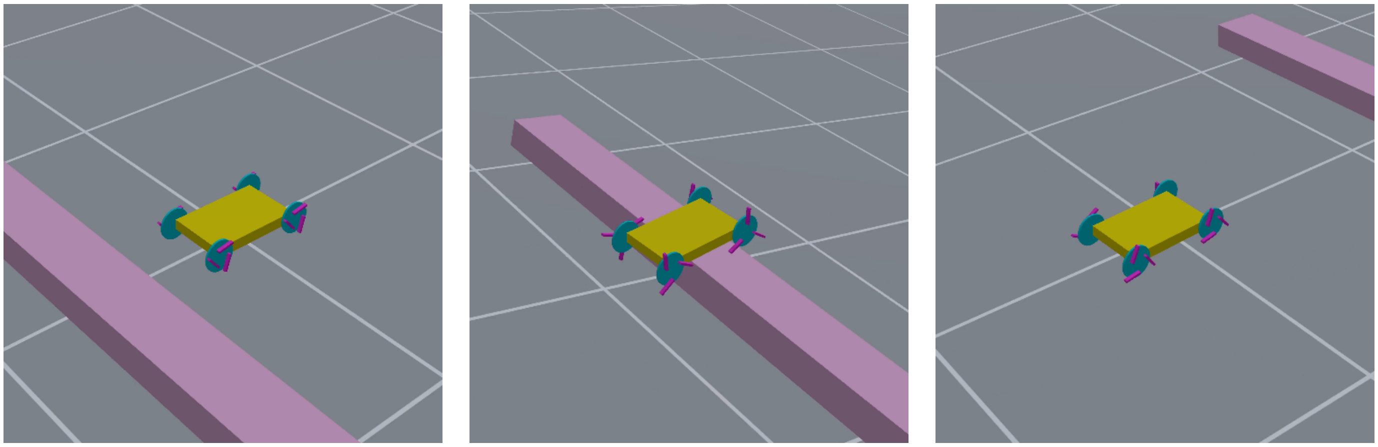
Fig. 5 shows the training results for each of the three experiments. We see a large jump in average reward for the policy trained without an obstacle. In early training steps, the policy prioritizes speed over retracting the extensions, and an obvious way to increase speed is to extend the extensions, which effectively increases the radius of each wheel. Around step 0.6×10^7 the policy learns to retract the extensions while maintaining the target velocity.

V. Conclusion

Our experiments indicate that proprioception alone may be sufficient for controlling the transformable wheels. The robot was able to climb over a single step obstacle using only proprioception, though it was better able to do so with a collision sensor.

We will continue this work by exploring a few different directions. First, we will optimize both the policy and the design of the robots. For example, we will optimize the number of legs per wheel, the shape of each leg, and the chassis dimensions. At the same time we will also optimize environment parameters to match real-world conditions. We also plan to improve reward design, accounting for more factors in evaluating performance of our controllers. In exploring these, we may take a co-design approach, co-optimizing wheel and leg parameters along with controller design to produce an optimal system.

Second, we will randomize the environment for each training episode. Specifically, the number of obstacles, their size, and



(a) Fully retracted extensions. (b) Fully extended extensions. (c) Partially retracted extensions.

Fig. 4: Our simulation environment, with the robot climbing over a step obstacle. The robot is unable to climb the obstacle without the extensions.

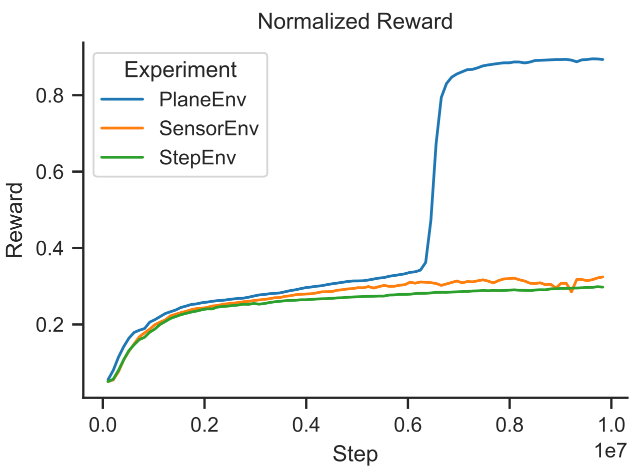


Fig. 5: Training results for our three experiments.

their location. This will help the policy generalize to different environments. We will also randomize the goal velocity of the robot to determine how it adapts to different speed requirements. Finally, we will set the robot to track randomized trajectories, rather than velocities, to increase its capabilities for deployment.

Third, we will explore more challenging terrain. Complex environments, such as those shown in Fig. 6, will make relying on proprioception alone more difficult. We will train the policy with randomized continuous terrains, implementing an adaptive curriculum to increase the difficulty of the terrains as the policy improves. If proprioception alone proves to be insufficient, we will also explore the use of cameras to better identify environmental features. This may include running simple computer vision models to gain simple insights about the terrain ahead, rather than attempting to observe full terrain maps. This will align with our goals of reducing sensor complexity and

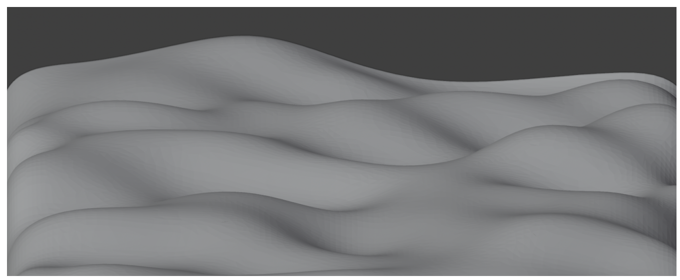


Fig. 6: Complex, continuous terrain for the robot to navigate. Terrain was generated using Blender.

primarily relying on proprioception. Finally, we will customize the policy training process to better suit the robot's design.

Acknowledgements

The authors would like to thank Tom Tang for his help developing the simulation environment.

References

[1] Ruixiang Cao, Jun Gu, Chen Yu, and Andre Rosendo. Omniwheg: An omnidirectional wheel-leg transformable robot. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5626–5631, 2022. doi: 10.1109/IROS47612.2022.9982030.

[2] Shen-Chiang Chen, Ke-Jung Huang, Wei-Hsi Chen, Shuan-Yu Shen, Cheng-Hsin Li, and Pei-Chun Lin. Quattroped: A Leg–Wheel Transformable Robot. *IEEE/ASME Transactions on Mechatronics*, 19(2):730–742, April 2014. ISSN 1941-014X. doi: 10.1109/TMECH.2013.2253615. URL https://ieeexplore.ieee.org/document/6508894. Conference Name: IEEE/ASME Transactions on Mechatronics.

[3] Xi Chen, Ali Ghadirzadeh, John Folkesson, Mårten Björkman, and Patric Jensfelt. Deep Reinforcement

Learning to Acquire Navigation Skills for Wheel-Legged Robots in Complex Environments. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3110–3116, October 2018. doi: 10.1109/IROS.2018.8593702. URL https://ieeexplore.ieee.org/document/8593702. ISSN: 2153-0866.

[4] Sebastian Höfer, Kostas Bekris, Ankur Handa, Juan Camilo Gamboa, Melissa Mozifian, Florian Golemo, Chris Atkeson, Dieter Fox, Ken Goldberg, John Leonard, C. Karen Liu, Jan Peters, Shuran Song, Peter Welinder, and Martha White. Sim2real in robotics and automation: Applications and challenges. *IEEE Transactions on Automation Science and Engineering*, 18(2):398–400, 2021. doi: 10.1109/TASE.2021.3064065.

[5] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. In *Rma: Rapid motor adaptation for legged robots*, 2021.

[6] Dae-Young Lee, Jae-Kyeong Kim, Chang-Young Sohn, Jeong-Mu Heo, and Kyu-Jin Cho. High–load capacity origami transformable wheel. *Science Robotics*, 6(53): eabe0201, 2021.

[7] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47): eabc5986, October 2020. doi: 10.1126/scirobotics.abc5986. URL https://www.science.org/doi/10.1126/scirobotics.abc5986. Publisher: American Association for the Advancement of Science.

[8] Inha Park, Sijun Ryu, Jeeho Won, Hyeungyu Yoon, SangGyun Kim, Hwa Soo Kim, and TaeWon Seo. Motion planning for 2-dof transformable wheel robots using reinforcement learning. *IEEE Robotics and Automation Letters*, 9(11):10193–10200, 2024. doi: 10.1109/LRA.2024.3469789.

[9] Ilija Radosavovic, Tete Xiao, Bike Zhang, Trevor Darrell, Jitendra Malik, and Koushil Sreenath. Real-world humanoid locomotion with reinforcement learning. *Science Robotics*, 9(89):eadi9579, April 2024. doi: 10.1126/scirobotics.adi9579. URL https://www.science.org/doi/10.1126/scirobotics.adi9579. Publisher: American Association for the Advancement of Science.

[10] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

[11] Or Simhon, Zohar Karni, Sigal Berman, and David Zarrouk. Overcoming obstacles with a reconfigurable robot using deep reinforcement learning based on a mechanical work-energy reward function. *IEEE Access*, 11:47681–47689, 2023. doi: 10.1109/ACCESS.2023.3274675.

[12] Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. *Robotics: Science and Systems*, 2025.

[13] Chuanqi Zheng and Kiju Lee. Wheeler: Wheel-leg reconfigurable mechanism with passive gears for mobile robot applications. In *2019 International Conference on Robotics and Automation*, pages 9292–9298, 2019. doi: 10.1109/ICRA.2019.8793686.

[14] İrem Mertyüz, Alper K. Tanyıldızı, Beyda Taşar, Ahmet B. Tatar, and Oğuz Yakut. Fuhar: A transformable wheel-legged hybrid mobile robot. *Robotics and Autonomous Systems*, 133:103627, 2020. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2020.103627. URL https://www.sciencedirect.com/science/article/pii/S092188902030467X.